Int. J. on Recent Trends in Engineering and Technology, Vol. 8, No. 1, Jan 2013

Modified Koblitz Encoding Method for ECC

[1]Ravi Kishore Kodali and Prof. Narasimha Sarma NVS
National Institute of Technology, Warangal
Department of E. and C. E., N.I.T., Warangal, India
Email: [1]kishore@nitw.ac.in

***Abstract*—Extensive use of Wireless Sensor Networks is giving rise to different types of threats in certain commercial and military applications. To protect the WSN data communication against various threats appropriate security schemes are needed. However, WSN nodes are resource constrained, with respect to limited battery energy, and limited computational and memory available with each WSN node. Hence, the security model to be used in WSN's should use minimal resources to the extent possible and it should also provide good security. Elliptic curve cryptography (ECC) is the best suited algorithm for WSNs, as it offers better security for smaller key sizes compared to the popular RSA algorithm. In ECC, encoding of message data to a point lying on the give Elliptic Curve is a major problem as the encoding consumes more resources. This paper provides a novel encoding procedure to overcome these problems to a large extent. This paper also describes implementation aspects of the proposed encoding and decoding methods.**

***Index Terms*— WSN, Cryptography, ECC, Koblitz's encoding**

I. Introduction

A wireless sensor network (WSN) consists of a large number of sensor nodes, which are randomly distributed over the given geographic region [1]. The sensors on these nodes carry out measurements of various physical phenomena related parameters and these measured values are collected, aggregated and forwarded towards the Base Station (BS). These WSN nodes are capable of self- organizing themselves to form a cooperative network. Various application areas, in which WSNs are made use of are: weather monitoring, indoor climate control, surveillance, forest fire detection and monitoring, structural health monitoring, medical diagnostics, disaster management and emergency response, ambient air monitoring [2]. WSN nodes are tiny in size, with limited amounts of memory, computational power and finite battery energy. Each node consists of one or more sensor types, a micro-controller, memory, a RF transceiver along with an antenna, and power electronics.

In both military and certain other applications, secure data need to reach the base station unaltered [3]. An intruder should not be able to decipher the secure data being sent between the BS and WSN nodes and among the nodes within WSN. Some of the possible attack types are: denial of service attack, Sybil attack, and attacks on information transmitted the information. To withstand against these attacks a proper security scheme is needed. It should provide confidentiality, integrity and authenticity, of all messages in the presence of various adversaries [3], [4]. In order to secure a data link, generally two types of cryptographic algorithms are used, namely, Symmetric key Cryptography (SKC) and asymmetric or Public key Cryptography (PKC). SKC algorithms such as advanced encryption standard (AES), International data encryption algorithm (IDEA) make use of the same key for both encryption (sender) and decryption (recipient) of the data. Each node should have been provided with an identical key before [3]. The AES, one of the SKC algorithms is more secure than PKC. Even though, the RSA algorithm is widely used PKC algorithm, it requires larger key sizes and hence thereby demanding more computational resources. These problems can be overcome by using the ECC algorithm. Even though SKC provides good security by utilizing fewer resources, key distribution is a major problem. If an adversary captures this key, then the security of the WSN is bound to be compromised. PKC algorithms, such as RSA, make use of two different keys, one key for encryption (sender) and another key and a different one for decryption (recipient). The RSA algorithm provides good security. However, it requires plenty of computational resources due to its large key size requirement and the associated exponential and modulo operations [5].

Elliptic curve cryptography (ECC) is another PKC algorithm, capable of providing a security level, comparable with that of RSA algorithm, with small key sizes. Table I compares the key lengths of ECC and RSA for equal security [1]. In ECC, in order to carry out encryption, a pre processing step, that is, encoding or mapping of a message data value to a point on the given elliptic curve is to be performed [6].

Table 1 : Security Level Comparison Of ECC And RSA

ECC key length	RSA key length
160	1024
224	2048
256	3072
384	7680

After carrying out decryption, a post processing step, that is, decoding or mapping the point on the given elliptic curve to its corresponding message data value, is to be performed. The commonly used encoding methods are: memory mapping and Koblitz's encoding [7]. The memory mapping method demands more memory, whereas, Koblitz's encoding method demands more computational resources and additional channel bandwidth. This paper proposes a novel encoding method, without any memory overheads. When compared with Koblitz's encoding, this method needs only fewer modulo operations and less channel overhead.

DOI: 01.IJRTET.8.1.54

II. Elliptic Curve Cryptography

ECC makes use of mathematical properties of the elliptic curves for both encryption and decryption. For cryptographic applications, ECC makes use of either the prime field or the binary field.The elliptic curve equation over prime field is represented by the equation (1). [8]

$$y^2 \bmod p = (x^3 + ax + b) \bmod p \quad (1)$$

The set of points satisfying the above elliptic curve equation and the point at infinity forms a group, represented by E_p (a, b). ECC uses the elements of this set, E_p (a, b), for encryption and decryption [9]. The basic operations involved in ECC are: addition of points and doubling of a point, scalar multiplication of a point.

A. Addition of points

Consider two distinct points on an elliptic curve, J and K, where J = (x_J, y_J) and K = (x_K, y_K). The addition of both these points results in another point on the given elliptic curve, L, where L= (x_L, y_L). x_L and y_L are derived by following mathematical expressions: [8]

$$x_L = (s^2 - x_L - x_K) \bmod p$$

$$y_L = (-y_J + s(x_J - x_L)) \bmod p,$$

where $s = ((y_J - y_K)/(x_J - x_K)) \bmod p \quad (2)$

B. Doubling of a point

Consider a point J on an elliptic curve, where J = (x_J, y_J), and $y_J \neq 0$. Then the doubling of this point J results in another point L on the given elliptic curve, where L = (x_L, y_L). x_L and y_L are derived by the following mathematical expressions: [9]

$$x_L = (s^2 - 2x_J) \bmod p$$

$$y_L = (-y_J + s(x_J - x_L)) \bmod p,$$

where $s = ((3x_J^2 + a)/2y_J) \bmod p \quad (3)$

C. Scalar multiplication

Scalar multiplication of point is obtained by multiplying a scalar with a point on the given elliptic curve. Scalar multiplication of a point can be achieved either by repeated addition of points operation alone or by combining repeated addition and doubling operations [10], [11]. The multiplication of a point p with a scalar k by using repeated additions is computed as follows:

k*P = P+P+P+P+P----------+P

(k times)

The multiplication of a point P with the scalar 12 by using repeated addition and doubling operations is computed iteratively as follows: 12*P = 2 (2 (2P + P))). Point multiplication can be accomplished by using different methods such as binary, Non Adjacent Form (NAF), window and comb method.

III. Discrete Logarthim Problem

The underlying security of ECC primarily relies on the difficulty level of the Elliptic Curve Discrete Logarithm Problem (ECDLP). Let P and Q be two points that lie on an elliptic curve such that k * P = Q, where k is a scalar. Given P and Q, it is computationally infeasible to obtain the value of k, if k is sufficiently large [10]and [14]. Then k is termed as discrete logarithm of Q to the base P. The ECDLP is proved to be more efficient than the RSA exponentiation problem. In ECC, scalar multiplication of a point is the primary one. While calculating addition and doubling operations, multiplicative inverse operation is required for finding the slope, s. The computation of multiplicative inverse operation involving more number of modulo operations consumes maximum CPU time. Let the number, a, be an element over prime field. The number b, is said to be the multiplicative inverse of the number, a, if it satisfies the following condition:

$$(a * b) \bmod p = 1$$

To compute multiplicative inverse we can make use of the following widely used algorithms: Exhaustive search, Almost Montgomery, and Extended Euclidean algorithms. Exhaustive search algorithm requires more computational resources. Hence, Extended Euclidean algorithm requiring fewer resources is used for this purpose.

In ECC, each user, sender and receiver chooses a private key, and the corresponding public key is derived from the user's private key by using scalar multiplication property of elliptic curves. An intruder with the knowledge of a public key cannot compute the corresponding private key because of the ECDLP. As ECC is a PKC algorithm, some of the parameters must be agreed upon by both the parties, the sender and the receiver and these are called domain parameters, a, b, p, G, n, where a, b, p are the parameters describing the given elliptic curve and G (generator point) and n (order of prime). The EC parameters are chosen such that for every message character within the entire message (ASCII) character set, there exists, a corresponding point on the EC. The order of the EC must be a prime, so that the shared key should not lie at the point of infinity. The G and n are computed as follows [8]:

A. Generator point (G)

The generator point, G, is a point on the elliptic curve, which is chosen such that the following [12] condition is met: n"G = O, where n is the large prime number and n is called order of the curve and O is called the infinity point [12]. The domain parameters are made public. Before sending any message, both the parties need to make their respective public keys public. Then both the parties compute shared key by their own private key and the other party's public key. This shared key is used for encryption and decryption purposes like in SKC.

B. Shared key calculation

Assume that the two parties involved in communication are Alice and Bob. The shared key is calculated as follows:

1. Either one, say Alice, chooses a point on the given elliptic

DOI: 01.IJRTET.8.1.54

curve as the generator point (G), satisfying the condition, n*G = O, where n is the largest prime number. The n - order of prime, G, generator point, along with the elliptic curve parameters are made known to each other by publishing these domain parameters. These domain parameters should be agreed upon mutually by both the parties (Alice and Bob), who intend to communicate securely [10].

2. Assuming Alice (sender) intends to send data, Alice selects her private key (senders private key), $n_A < n$ and computes her public key, P_A, which is a point on the elliptic curve, using $P_A = n_A * G$.
   On the other side, Bob (receiver) selects his private key (receiver's private key), $n_B < n$ and computes his public key P_B, which is also a point on the elliptic curve, using $P_B = n_B * G$
   An intruder can know only public keys of both the sender and the receiver. The intruder cannot compute the corresponding private keys from this knowledge alone because of the ECDLP [8].
3. Alice computes the shared key point using Bob's public key and her own private key using the following equation:

$$S_K = P_B * n_A = n_A * n_B * G.$$

In the similar manner, Bob also computes simultaneously, the shared key point using Alice's public key and his own private key using the following equation:

$$S_K = P_A * n_B = n_A * n_B * G$$

Now both Alice and Bob have the same shared key. Here after, Alice and Bob can start using this shared key to encrypt and decrypt any of the messages [7].

IV. Encryption And Decryption [10]

A. Encryption steps

1. Alice encodes or maps a message value to a point, (P_M), on the elliptic curve

2. Then Alice encrypts this message point, (P_M), to obtain the corresponding cipher point, (P_C), lying on the same elliptic curve using the following equation:

$$P_C = [P_M + S_K]$$

B. Decryption steps

1. Upon receiving the cipher point, (P_C), Bob subtracts the shared key (S_{-K}) from the cipher point, (P_C), to get the encoded message point (P_M), the same is also expressed by the equation, $P_M = P_C * S_K$. The additive inverse of (S_K), a point (x, y) on the elliptic curve is another point on the same elliptic curve (x, p " y) and the same additive inverse is denoted by ($-S_K$).

2. Then, Bob decodes or maps the message point, (P_M), into the corresponding message value.

In ECC, encryption and decryption steps are performed over the points on the given elliptic curve, whereas general input comprises of ASCII values representing alpha-numeric. Hence, before carrying out the encryption, a pre-processing step, encoding or mapping of a message data value to a point on the given elliptic curve is to be performed [6].

After carrying out decryption, a post processing step, that is, decoding or mapping the point on the given elliptic curve to its corresponding message data value, is to be performed. The commonly used encoding methods are: memory mapping and Koblitz's encoding [7]. The memory mapping method demands more memory, whereas Koblitz's encoding method demands more computational resources and additional channel bandwidth.

V. Encoding Techniques

A. Memory mapping

Consider a set of 128 ASCII symbols. Any input message has to make use of some of these symbols from this set. Now an elliptic curve, with a minimum of 128 points needs to be selected. This is required to map each ASCII symbol to a point on the elliptic curve distinctly. The 128 elliptic curve points are stored in a memory, for which the input message ASCII value acts as an index. By providing an index, the corresponding stored value can be retrieved. This retrieved value corresponds to a point on the elliptic curve. While decoding, the elliptic curve point value is matched against each of the entries in the memory. Wherever the match occurs, the corresponding index value is treated as its message ASCII value. It can be noticed that this decoding consumes more time. The memory mapping method is unsuitable in WSN applications, as the contents of the memory device can be read, if any node is physically captured by an intruder. [7]

B. Koblitz's encoding method

Choose an elliptic curve and its associated auxiliary base parameter, k, in such a way that there exists at least one point within the range of x values given by [(m* k) + 1 to (m * k) + k], where m represents the ASCII value of the message. In order to encode or map a message value, m, try to solve for y by substituting x = (m * k) + 1 in the chosen elliptic curve equation. If the y value is obtained, then take the corresponding encoded point as [(m * k) + 1, y]. If the y value cannot be obtained, then increment the x value by one (m * k) + 2 and then, try to obtain y. The same can be continued up to x = (m * k) + k. For every m in the message character set, the above procedure is repeated. The maximum, among each of these values of k, is considered as the auxiliary parameter for the entire message character set. In order to decode the decrypted point lying on the elliptic curve, (x, y), the operation (x - 1)/k is performed. The integer quotient of this division operation represents the ASCII value of the message m. In Koblitz's encoding, few extra bits need to be transmitted as the message, m, is multiplied by auxiliary base parameter, k. For the ASCII set of 128 characters and k = 10, the encoding overhead is 4- bits. Further the number of required computations also increases.

C. Modified Koblitz's encoding procedure

It can be observed from the non-singular elliptic curve equation by choosing b = 0, there exists a class of elliptic

DOI: 01.IJRTET.8.1.54

curves, represented by the non-singular elliptic curve equation (2).

$$y^2 \ mod \ p = (x^3 + ax) \ mod \ p \quad (4)$$

The equation (4) satisfies the following two mathematical properties:

Property- 1:

Consider the l.h.s. of the equation

$$y^2 \ mod \ p = (x^3 + ax) \ mod \ p$$

Let y = k, where k = {0,…, p-1}.

If (y^2 mod p) = r, then for any value of y in the prime field, the resultant (y^2 mod p) $\neq$(p-r).

This property -1 is applicable for those primes satisfying the condition p = (4*i) – 1, where i is a positive integer.

Property- 2:

Consider the r.h.s. of the equation

$$y^2 \ mod \ p = (x^3 + ax) \ mod \ p$$

Let x = l, where l = {0,…,p-1}

If , then for the value x = (p-l), the resultant of .

$$(x^3 + ax) \ mod \ p = (p - t).$$

Proof for the Property 2:

Consider the r.h.s. of the elliptic curve equation

$$y^2 \ mod \ p = (x^3 + ax) \ mod \ p$$

for any value, x = l, where l = {0,…,p-1}

$$(x^3 + ax) \ mod \ p = (l^3 + al) \ mod \ p = l^3 + al$$

Let $(l^3 + al) = t, \ for \ x = (p - l),$

$$(x^3 + ax) \ mod \ p = [(p - l)^3 + a * (p - l)] \ mod \ p$$

$$= (p^3 - l^3 - 3pl(p - l) + ap - al) \ mod \ p$$

$$= (-l^3 - al) \ mod \ p = p - (l^3 + al) = (p - t)$$

The property-1 is illustrated with an example. Let the prime value p = 31. All the possible (y^2 mod p) values are computed by varying the values of y in the prime field, {0, 1, , 30}, and given the same in Table II. Let these computed resulting values be a quadratic residue set (Q_{31}). Q_{31} = (0, 1, 2, 4, 5, 7, 8, 9, 10, 14, 16, 18, 19, 20, 25, 28), where Q_{31} are called Quadratic residue set.

From the Q_{31}, it can be observed that for y = 5, (y^2 mod 31) = (52 mod 31) = 25. (p - r) = (31 - 25) = 6, is not found in the Q_{31}. The same can be verified for the remaining values of y in the prime field. The proposed encoding procedure makes use of both property -1 and property- 2 to map a message character to a point on the given elliptic curve.

Consider the elliptic curve equation (2) over the prime field. Let the message ASCII value is m. In order to encode this message value, m is substituted in the elliptic curve equation in place of x (x = m). If the value lies in the corresponding quadratic residue set, then there exist two possible values for y. Let these two be y1 and y2. Now the first point (m, y1) is used as the encoded point on the elliptic curve. If the value, r, is not present within the quadratic residue set, then substitute x = (p”m). Based on the property -2 the following results in:

$$[(p - m)^3 + a(p - m)] \ mod \ p = (p - r)$$

Based on the property -1, this value (p – r) needs to necessarily be present in the corresponding quadratic residue set. This implies that there exist two possible values for y. Let these two also be y_1 and y_2. Now, the second point ((p – m), y_2) is used as the encoded point on the elliptic curve. The same is also presented in the flow chart 1. In order to decode the decrypted point lying on the elliptic curve, (x, y), if y < p/2, then the value x represents the ASCII value of the message. Otherwise, (p-x) represents the ASCII value of the message.

TABLE II. QUADRATIC RESIDUE SET COMPUTATION

Y^2 mod 31	$[31 - Y]^2$ mod 31	Result
1^2 mod 31	30^2 mod 31	1
2^2 mod 31	29^2 mod 31	4
3^2 mod 31	28^2 mod 31	9
4^2 mod 31	27^2 mod 31	16
5^2 mod 31	26^2 mod 31	25
6^2 mod 31	25^2 mod 31	5
7^2 mod 31	24^2 mod 31	18
8^2 mod 31	23^2 mod 31	2
9^2 mod 31	22^2 mod 31	19
10^2 mod 31	21^2 mod 31	7
11^2 mod 31	20^2 mod 31	28
12^2 mod 31	19^2 mod 31	20
13^2 mod 31	18^2 mod 31	14
14^2 mod 31	17^2 mod 31	10
15^2 mod 31	16^2 mod 31	8

The proposed encoding method, as given by Fig.1 is illustrated with an example. Let the prime value be p, p = 31 and a = “1 then the possible valid points on this elliptic curve are computed and arranged as shown in Table III.

TABLE III. COMPUTATION EXAMPLE

X	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$(x^3 - x)$ mod 31	0	0	6	24	29	27	24	26	8	7	29	18	11	14	2	12
$Y^2 Q_{31}$	Y	Y	N	N	N	N	N	N	Y	Y	N	Y	N	Y	Y	N
$Y1$	0	0							15	10		7		13	8	
$Y2$	0	0							16	21		24		18	23	

X	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
$(x^3 - x)$ mod 31	0	25	7	2	4	7	5	23	24	2	13	20	17	29	19
$Y^2 Q_{31}$	Y	Y	Y	Y	Y	Y	Y	N	N	Y	N	Y	N	N	Y
$Y1$	0	5	10	8	2	10	6			8		12			9
$Y2$	0	26	21	23	29	21	25			23		19			22

Consider the input message value of 8. From Table III, it can be observed that there exist, two possible y’s, y_1 = 15 and y_2 = 16. Then the first point, (8, 15) is used as the encoded point for the input message value of 8. By considering another input value of 10, it can be seen that no possible y value exists. However, based on the properties 1 and 2, there exist two possible y’s, for x = (p-l) = (31 - 10) = 21, y_1 = 8 and

DOI: 01.IJRTET.8.1.54

y_2 = 23. Then the second point (21, 23) is as the corresponding encoded point for the input message value of 10. The same is valid for all the remaining input values as well.
In this encoding procedure, no auxiliary base parameter is used. As compared to Koblitz's encoding method, the encoding overhead is removed. Table IV gives the resource utilization comparison of both the Koblitz's encoding method and the proposed modified Koblitz's encoding method.

TABLE IV. RESOURCE COMPARISON

Encoding type	Input Data [bits]	Encoded output size [bits]	No. of Modulo operations	Security [bits]
Koblitz encoding	7	12	5672	11
Modified encoding	11	12	465	9

VI. CONCLUSIONS

ECC, using memory mapping encoding method is unsuitable for resource constrained applications, such as, WSN applications. ECC, using Koblitz's encoding method consumes more channel bandwidth and computational resources as a consequence of encoding overheads. However, the proposed encoding method for ECC saves about 90 percent of computational resources required for Koblitz's encoding method. It also reduces the channel bandwidth overhead. However, the security level is reduced by 2- bits. ECC, with the proposed encoding method is well suited for WSN applications having 32- bit key size requirements.

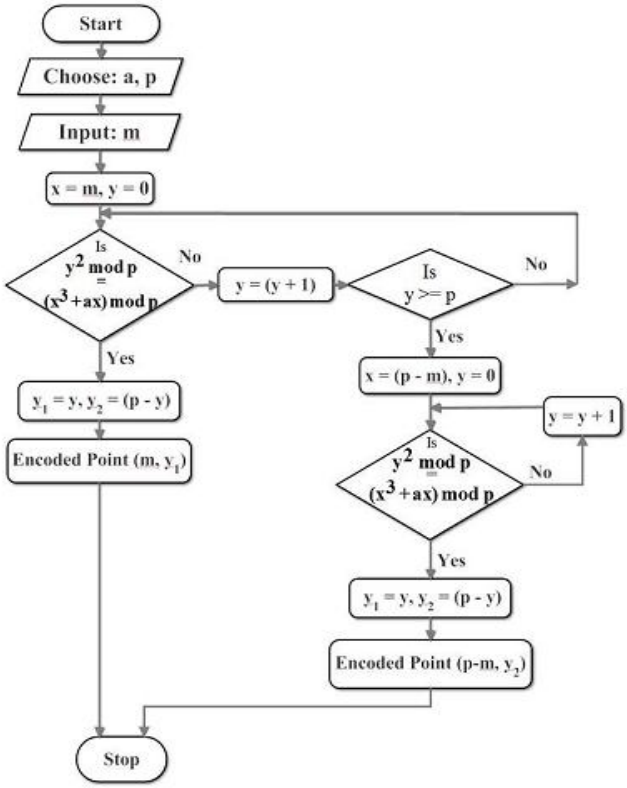

Fig. 1. Modified Koblitz Encoding

REFERENCES

[1] M. Ahmed, S. Alam, N. Qureshi, and I. Baig, "Security for wsn based on elliptic curve cryptography," in *Computer Networks and Information Technology (ICCNIT), 2011 International Conference on*. IEEE, 2011, pp. 75–79.

[2] A. Praveena, S. Devasena, and K. Chelvan, "Achieving energy efficient and secure communication in wireless sensor networks," in *Wireless and Optical Communications Networks, 2006 IFIP International Conference on*. IEEE, 2006, pp. 5–pp.

[3] Y. Qian, K. Lu, and D. Tipper, "A design for secure and survivable wireless sensor networks," *Wireless Communications, IEEE*, vol. 14, no. 5, pp. 30–37, 2007.

[4] I. Fernandez and W. Subbarao, "Encryption based security for isdn communication: Technique and application," in *Southeastcon '94. 'Creative Technology Transfer-A Global Affair', Proceedings of the* IEEE, 1994, pp. 70–72.

[5] A. kumar, D. S. Tyagi, M. Rana, N. Aggarwal, and P. Bhadana, "A comparative study of public key cryptosystem based on ecc and rsa," *Wireless Communications, IEEE*, vol. 3, no. 5, pp. 1904–1909, 2011.

[6] S. Kadir, A. Sasongko, and M. Zulkifli, "Simple power analysis attack against elliptic curve cryptography processor on fpga implementation," in *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*. IEEE, 2011, pp. 1–4.

[7] P. Bh, D. Chandravathi, and P. Roja, "Encoding and decoding of a message in the implementation of elliptic curve cryptography using koblitzs method," *International Journal on Computer Science and Engineering*, vol. 2, pp. 1904–1907, 2010.

[8] R. Pateriya and S. Vasudevan, "Elliptic curve cryptography in constrained environments: A review," in *Communication Systems and Network Technologies (CSNT), 2011 International Conference on*. IEEE, 2011, pp. 120–124.

[9] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *Wireless Communications, IEEE*, vol. 11, no. 1, pp. 62–67, 2004.

[10] X. Huang, P. Shah, and D. Sharma, "Protecting from attacking the man-in- middle in wireless sensor networks with elliptic curve cryptography key exchange," in *Network and System Security (NSS), 2010 4th International Conference on*. IEEE, 2010, pp. 588–593.

[11] X. Sun and M. Xia, "An improved proxy signature scheme based on elliptic curve cryptography," in *Computer and Communications Security, 2009. ICCCS'09. International Conference on*. IEEE, 2009, pp. 88–91.

[12] A. Rahuman and G. Athisha, "Reconfigurable architecture for elliptic curve cryptography," in *Communication and Computational Intelligence (INCOCCI), 2010 International Conference on*. IEEE, 2010, pp. 461– 466.

DOI: 01.IJRTET.8.1.54